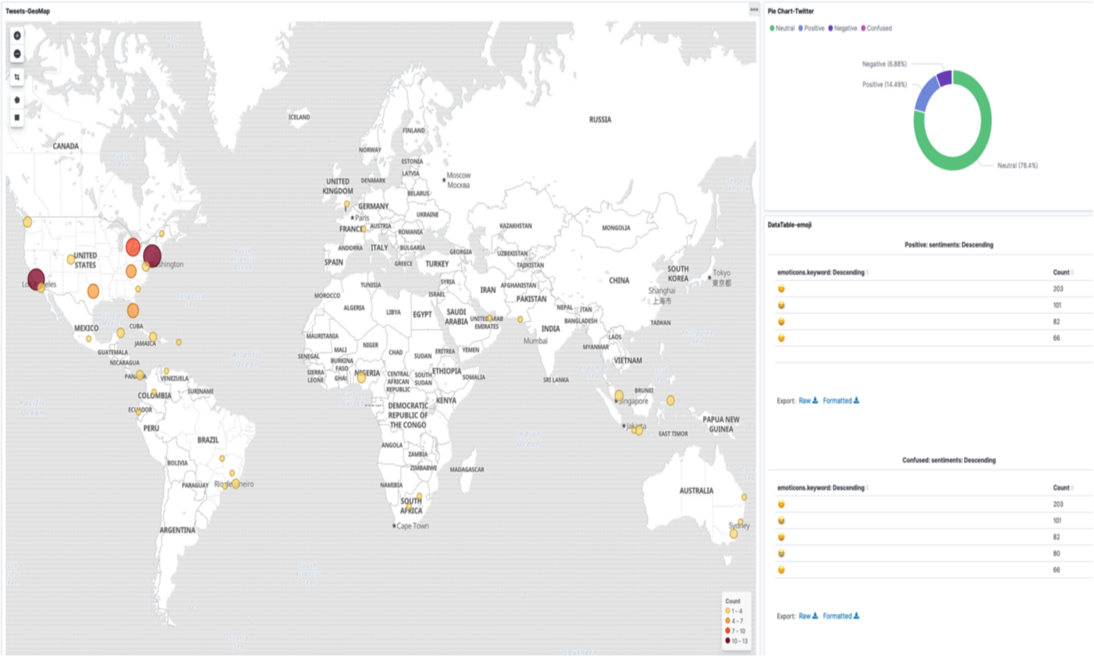
TwitterAPI with OpenSearch

> using knowledge of Cloud Computing Course which tough by professor Unubold Tumenbayar From Streaming Data to COVID-19 Twitter Analysis: Using AWS Lambda, Kinesis Firehose and Elasticsearch

From Streaming Data to COVID-19 Twitter Analysis: Using AWS Lambda, Kinesis Firehose and Elasticsearch

Unleash the power of AWS and Elasticsearch



DataTable-emoji

Positive: sentiments: Descending

emoticons.keyword: Descending	Count
😊	203
😂	101
😃	82
😉	66

Export: Raw Formatted

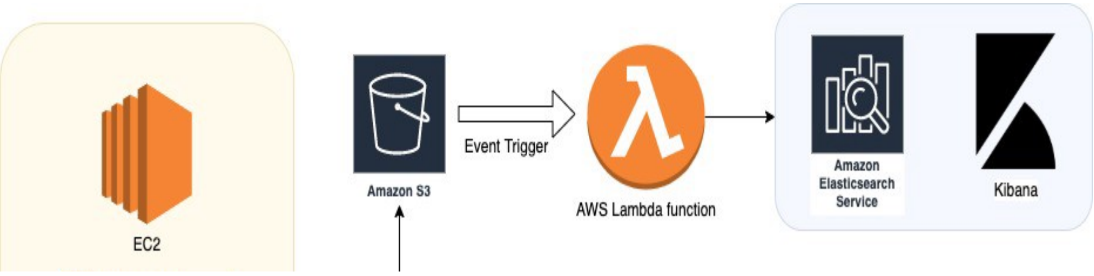
Confused: sentiments: Descending

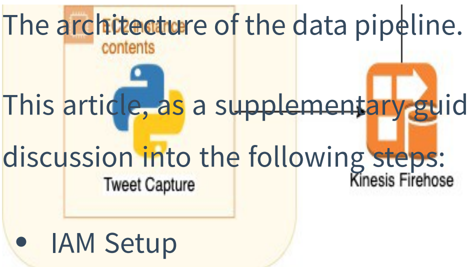
emoticons.keyword: Descending	Count
😊	203
😂	101
😃	82
😭	80
😉	66

Export: Raw Formatted

The dashboard of analyzing the real-time tweets.

The data pipeline follows the fashion of near real-time streaming, from digesting live Twitter data, to visualization. Most of the fragments to assemble the puzzle are from the AWS family: **AWS Kinesis Firehose**, **AWS S3 Bucket**, **AWS Lambda**, and **Amazon Elasticsearch Service**. The architecture is like this:





The architecture of the data pipeline.

This article, as a supplementary guideline to the *reference article*, will separate the discussion into the following steps:

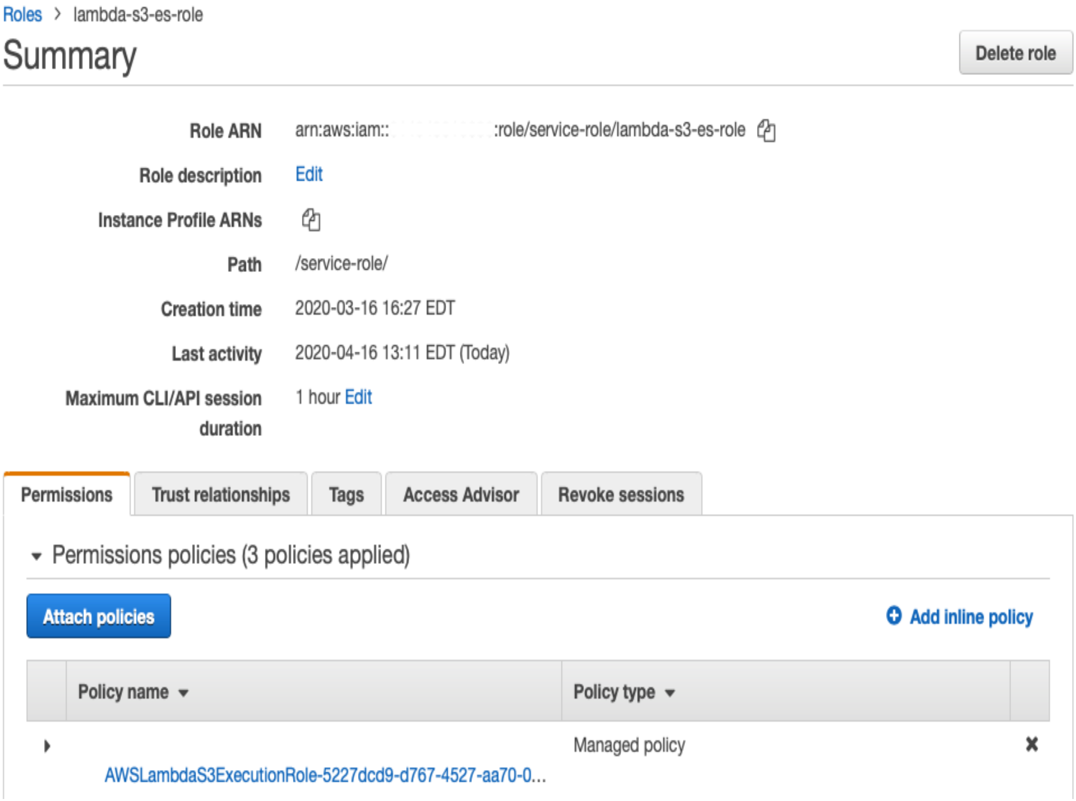
- IAM Setup
- Creating the Amazon Elasticsearch Service cluster
- Configuring the AWS Kinesis Firehose and S3
- Creating the AWS Lambda function
- Code packaging and changes
- Kibana visualization and Twitter analysis

The code for **the Lambda function** and **Twitter capture program** has been uploaded in my public repo.

IAM Setup

Before building the data pipeline, you need to have an AWS account and Twitter API keys and access tokens, which is also mentioned in the prerequisites of *the reference article*. Besides that, IAM roles are critical and must be set up correctly. Two roles are needed:

- Kinesis Firehose needs an IAM role with granted permissions to deliver stream data, which will be discussed in the section of Kinesis and S3 bucket.
- AWS Lambda needs permissions to access the S3 event trigger, add CloudWatch logs, and interact with Amazon Elasticserch Service.



Summary

Role ARN arn:aws:iam::[redacted]:role/service-role/lambda-s3-es-role

Role description Edit

Instance Profile ARNs

Path /service-role/

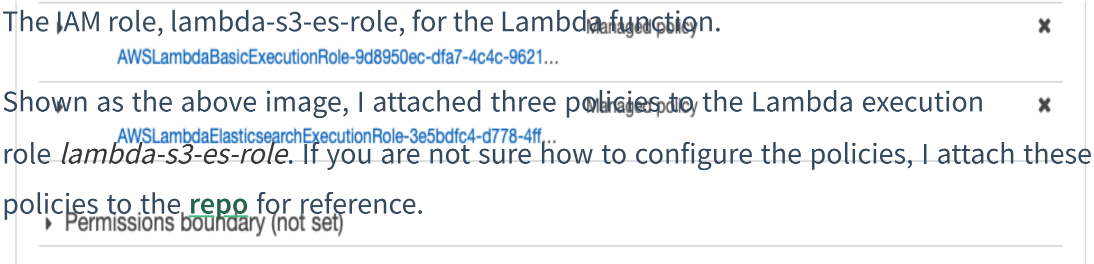
Creation time 2020-03-16 16:27 EDT

Last activity 2020-04-16 13:11 EDT (Today)

Maximum CLI/API session duration 1 hour Edit

Permissions policies (3 policies applied)

Policy name	Policy type
AWSLambdaS3ExecutionRole-5227dcd9-d767-4527-aa70-0...	Managed policy



Managed policy
AWSLambdaBasicExecutionRole-9d8950ec-dfa7-4c4c-9621...

Managed policy
AWSLambdaElasticsearchExecutionRole-3e5bdfc4-d778-4ff...

Permissions boundary (not set)

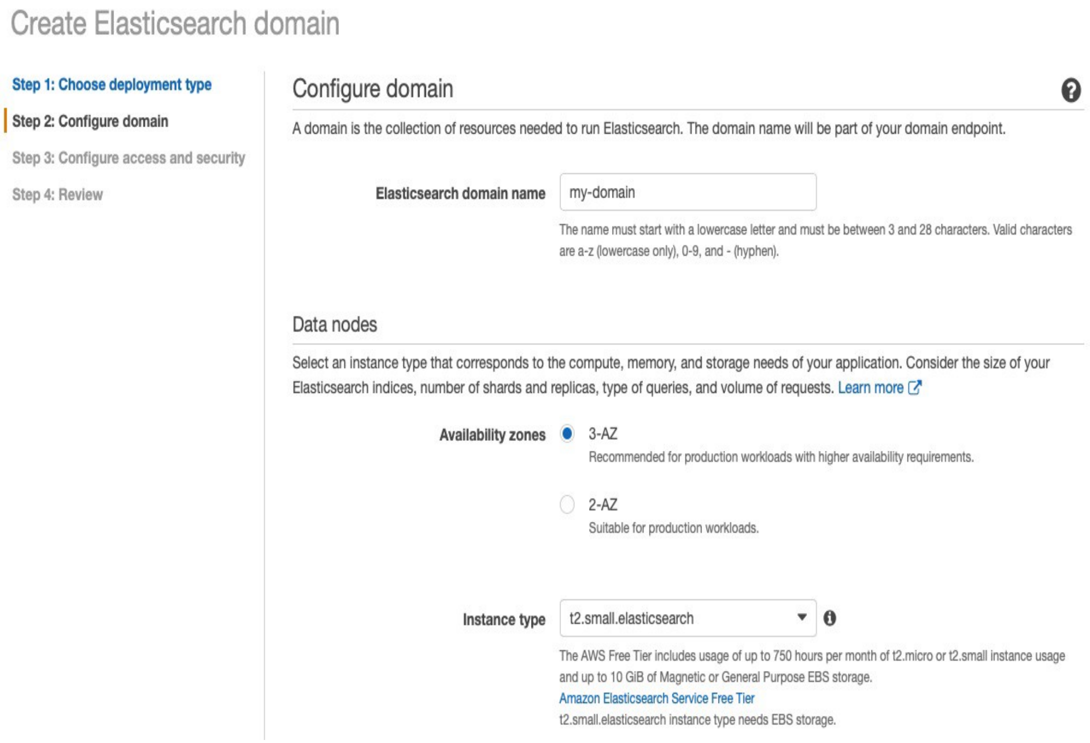
The IAM role, lambda-s3-es-role, for the Lambda function.

Shown as the above image, I attached three policies to the Lambda execution role *lambda-s3-es-role*. If you are not sure how to configure the policies, I attach these policies to the repo for reference.

Creating the Amazon Elasticsearch Service (ES) cluster

I assume that readers have followed through the steps in the *reference article* to create an Amazon ES domain at Amazon ES home page. For free-tier users, they can choose instance types as t2.micro or t2.small, and earn free 750 hours usage of Amazon ES. A few points need to note when creating the ES domain:

- No need to set up “dedicated master nodes”.
- As a demo project, I choose public access in “network configuration”.



Create Elasticsearch domain

Step 1: Choose deployment type
Step 2: Configure domain
Step 3: Configure access and security
Step 4: Review

Configure domain

A domain is the collection of resources needed to run Elasticsearch. The domain name will be part of your domain endpoint.

Elasticsearch domain name: my-domain

The name must start with a lowercase letter and must be between 3 and 28 characters. Valid characters are a-z (lowercase only), 0-9, and - (hyphen).

Data nodes

Select an instance type that corresponds to the compute, memory, and storage needs of your application. Consider the size of your Elasticsearch indices, number of shards and replicas, type of queries, and volume of requests. Learn more

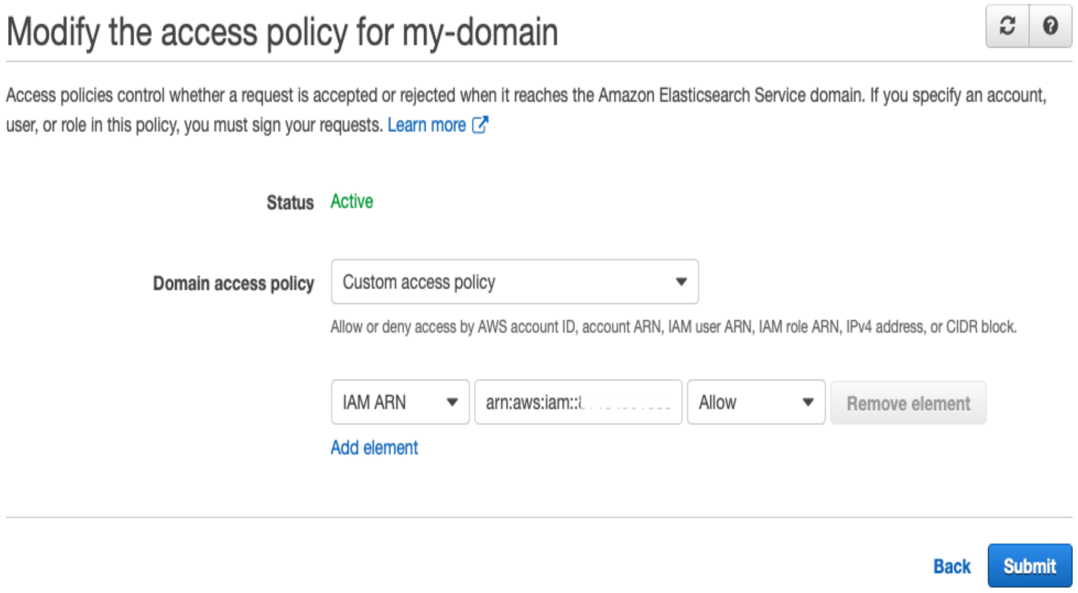
Availability zones
● 3-AZ
Recommended for production workloads with higher availability requirements.
○ 2-AZ
Suitable for production workloads.

Instance type: t2.small.elasticsearch

The AWS Free Tier includes usage of up to 750 hours per month of t2.micro or t2.small instance usage and up to 10 GiB of Magnetic or General Purpose EBS storage.
Amazon Elasticsearch Service Free Tier
t2.small.elasticsearch instance type needs EBS storage.

Here is the thing. When configuring “Access policy”, choosing “customer access policy”, you need to add the following policy:

1. Select “ARN”, and allow the lambda execution role *lambda-s3-es-role* to access ES service.



Modify the access policy for my-domain

Access policies control whether a request is accepted or rejected when it reaches the Amazon Elasticsearch Service domain. If you specify an account, user, or role in this policy, you must sign your requests. Learn more

Status: Active

Domain access policy: Custom access policy ▾

Allow or deny access by AWS account ID, account ARN, IAM user ARN, IAM role ARN, IPv4 address, or CIDR block.

IAM ARN ▾ | arn:aws:iam:: | Allow ▾

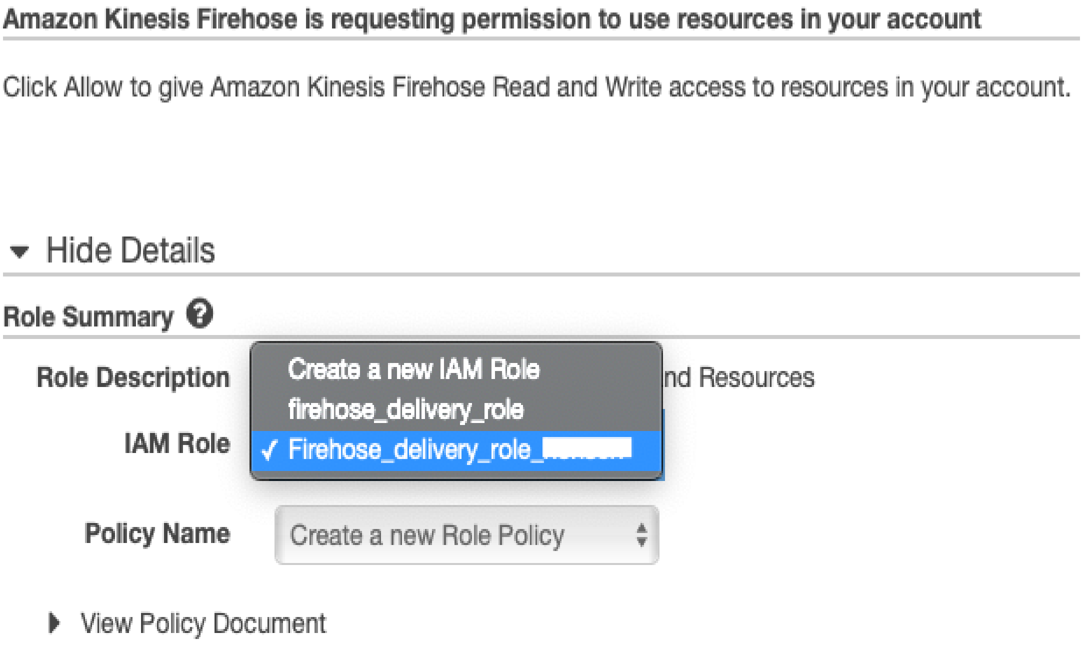
I leave a question here: **are the settings correct?** We will test it later.

Configuring the AWS Kinesis Firehose and S3

Different from the *reference article*, I choose to create a Kinesis Firehose at the Kinesis Firehose Stream console. The steps are simple:

- Fill a name for the Firehose Stream
- Source: Direct PUT or other sources
- Destination: an S3 bucket, which is used to store data files (actually, tweets). Here you can choose an S3 bucket you have created or create a new one on the fly.
- Permissions.

As mentioned in the **IAM Section**, a Firehose Stream needs IAM roles to contain all necessary permissions. Click “Create new or choose”, and choose to “create a new IAM Role”, or use an existing one. The default policy would be attached and should meet the need.



Amazon Kinesis Firehose is requesting permission to use resources in your account

Click Allow to give Amazon Kinesis Firehose Read and Write access to resources in your account.

Kinesis Data Firehose Stream (KDF) and Kinesis Data Stream (KDS) may confuse people sometimes. KDF has extra features when delivering stream data. Source data is allowed to be transformed through a Lambda function during delivery to destination. My other post covers the usage of KDF.

Creating the AWS Lambda function

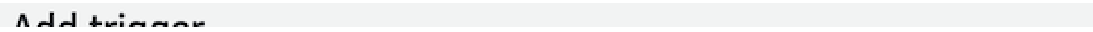
AWS Lambda plays a central role in this pipeline. We will create a Lambda function to do the following jobs:

- Once a new data file created in the target S3 bucket, the Lambda function would be triggered.
- Data would be parsed with a designated structure, which agrees with the mapping for documents.
- The data would be loaded into the ES cluster.

To implement such a Lambda function at one stroke is hard. Dividing the cumbersome procedure into smaller steps, I first need to set up the Lambda environment correctly.

Create a function at AWS Lambda home page:

- Choosing Python 3.7 runtime.
- Choosing *lambda-s3-es-role* as the execution role.
- Keeping memory at 128 MB and set timeout as 2 min.
- Adding a trigger for S3. If any new file comes to the S3 bucket, the Lambda function would receive the event and get invocated.



Add trigger

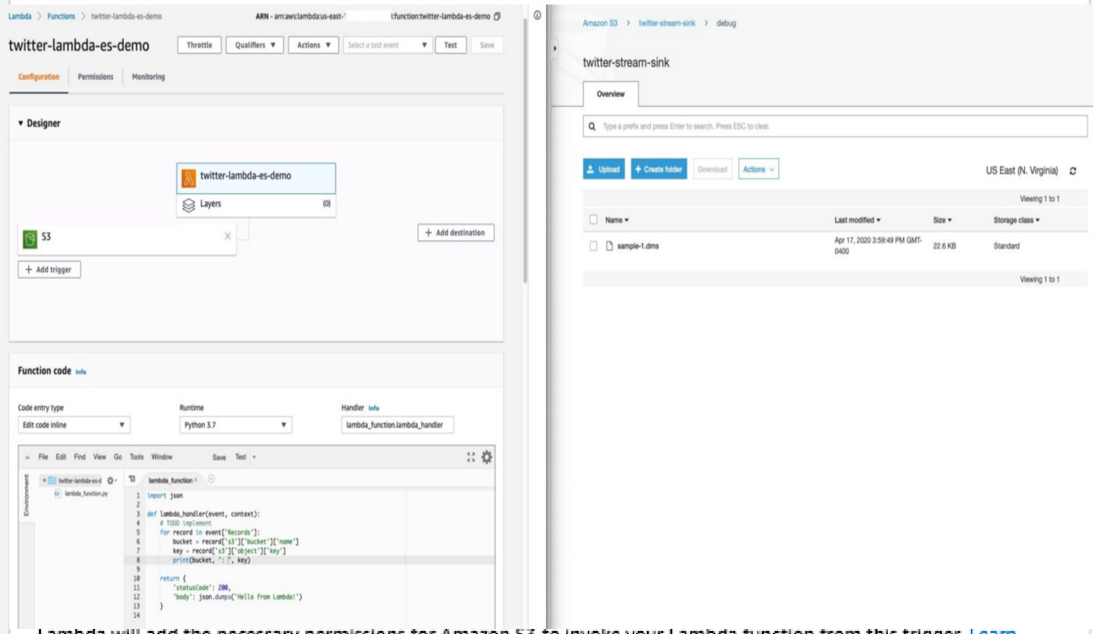
Add trigger

Create an S3 trigger for Lambda function, note that prefix “debug” is used for debugging and can be substituted at your needs.

Trigger configuration

Now, we can test if the Lambda function could react to the S3 event. With sample code and configured Handler, we put a file into the S3 bucket *twitter-stream-sink*.

S3
aws storage



Code entry type: Edit code inline
Runtime: Python 3.7
Handler: lambda_function.lambda_handler

```
import json

def lambda_handler(event, context):
    # TODO implement
    for record in event['Records']:
        bucket = record['s3']['bucket']['name']
        key = record['s3']['object']['key']
        print(bucket, ": ", key)

    return {
        'statusCode': 200,
        'body': json.dumps('Hello from Lambda!')
    }
```

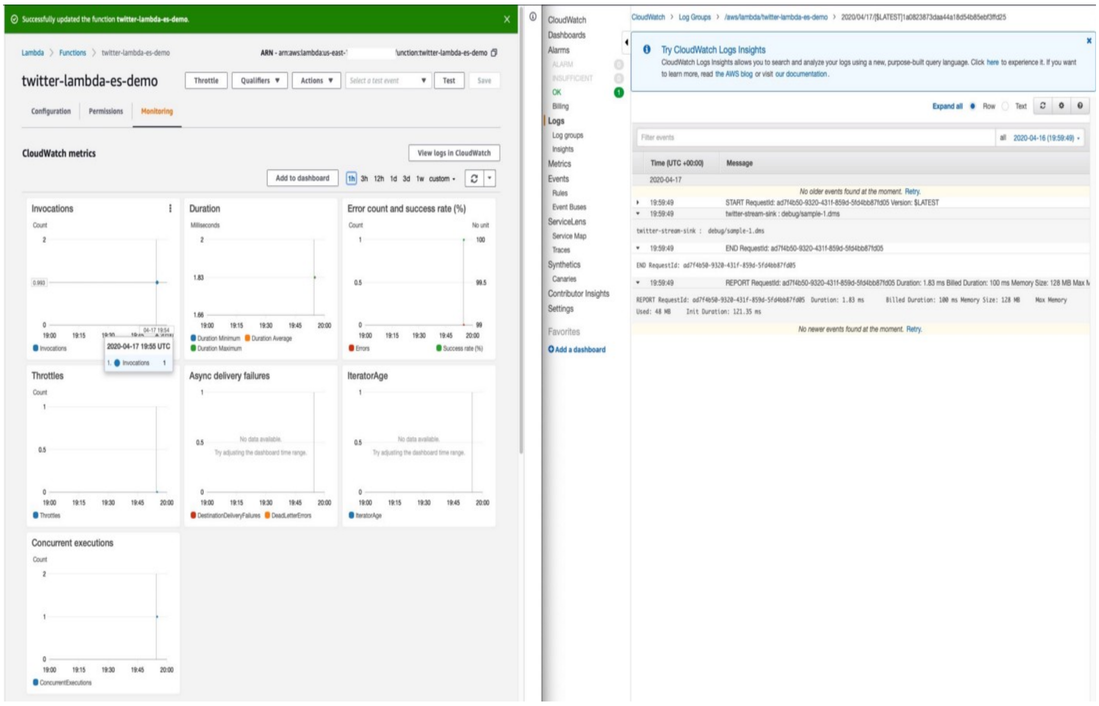
Lambda will add the necessary permissions for Amazon S3 to invoke your Lambda function from this trigger. Learn more about the Lambda permissions model.

Create a Lambda function and put a test file into the S3 bucket. Note that the Handler name should match with the entry function in Function code.

☑ Enable trigger

Enable the trigger now, or create it in a disabled state for testing (recommended).

On the “Monitoring” tab on the Lambda function panel, there is one dot appearing on the metrics graphs. Clicking “View logs in CloudWatch”, we have the *CloudWatch* log for this invocation, and the log prints the source S3 bucket and the file name which we just put in.



```
Time (UTC +00:00)   Message
2020-04-17
                    No older events found at the moment. Retry.
19:59:49            START RequestId: ad7f4b50-9320-431f-859d-5fd4bb87fd05 Version: $LATEST
19:59:49            twitter-stream-sink : debug/sample-1.dms
twitter-stream-sink :  debug/sample-1.dms
19:59:49            END RequestId: ad7f4b50-9320-431f-859d-5fd4bb87fd05
END RequestId: ad7f4b50-9320-431f-859d-5fd4bb87fd05
19:59:49            REPORT RequestId: ad7f4b50-9320-431f-859d-5fd4bb87fd05 Duration: 1.83 ms Billed Duration: 100 ms Memory Size: 128 MB Max M
REPORT RequestId: ad7f4b50-9320-431f-859d-5fd4bb87fd05  Duration: 1.83 ms    Billed Duration: 100 ms Memory Size: 128 MB    Max Memory
Used: 48 MB    Init Duration: 121.35 ms
                    No newer events found at the moment. Retry.
```

The left window shows a dot in metrics graphs, denoting an invocation on the Lambda function. The right window shows the CloudWatch log details.

Code packaging and changes

The *reference article* was published several years ago, so the project code needs updates.

The project code can be downloaded from the **repo**. The code directory contains 4 python files: *config.py*, *myhandle.py*, *tweet_utils.py*, *twitter_to_es.py*. To add necessary libraries into the project folder, just need to type the command:

```
pip install library_name -t .
```

The libraries needed to import to the project directory are such as:

```
requests
requests_aws4auth
elasticsearch
textblob=0.15
```

The Lambda function accepts a code package with zip format. Packaging the code directory with libraries needs this command:

```
zip -r ../your_package_name.zip * -x "*.git*"
```

The tutorial about the Lambda deployment package can be found in **AWS Lambda document**.

Now let us take a glimpse at each python file:

myhandle.py

- Serve as the entry point for the Lambda function.
- Parse the event information, and obtain S3 file content in JSON format.

twitter_to_es.py

- Add indices and mappings to the ES cluster.
- Adopt the **bulk** way to load parsed data into the ES cluster.
- Authorize the requests sent to the ES cluster.

tweet_utils.py

- Act as a helper module.
- Parse tweets into the structured dictionary.
- Analyze the sentiments over tweets using TextBlob.

config.py

- Act as the shared configuration.

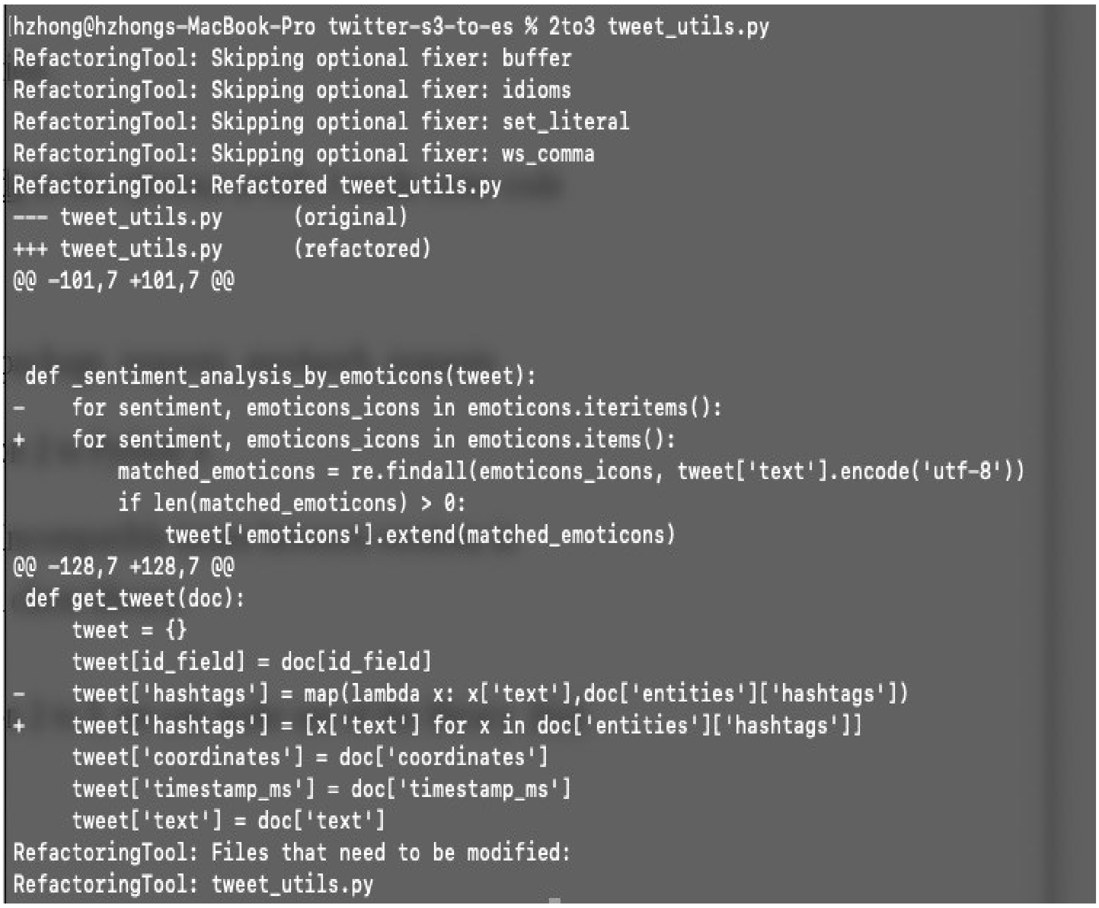
Compared with the original code in the *reference article*, I made some code changes:

1. Add extra libraries into the package, requests_aws4auth, requests.

2. Port source code from Python 2 to Python 3.

3. Fix the bugs because Elasticsearch and its Python client library have the incompatible issues with previous versions.

To port source code from Python 2 to 3, we can use the library `2to3` .



```
hzhong@hzhongs-MacBook-Pro twitter-s3-to-es % 2to3 tweet_utils.py
RefactoringTool: Skipping optional fixer: buffer
RefactoringTool: Skipping optional fixer: idioms
RefactoringTool: Skipping optional fixer: set_literal
RefactoringTool: Skipping optional fixer: ws_comma
RefactoringTool: Refactored tweet_utils.py
--- tweet_utils.py      (original)
+++ tweet_utils.py      (refactored)
@@ -101,7 +101,7 @@


 def _sentiment_analysis_by_emoticons(tweet):
-    for sentiment, emoticons_icons in emoticons.iteritems():
+    for sentiment, emoticons_icons in emoticons.items():
         matched_emoticons = re.findall(emoticons_icons, tweet['text'].encode('utf-8'))
         if len(matched_emoticons) > 0:
             tweet['emoticons'].extend(matched_emoticons)
@@ -128,7 +128,7 @@
 def get_tweet(doc):
     tweet = {}
     tweet[id_field] = doc[id_field]
-    tweet['hashtags'] = map(lambda x: x['text'],doc['entities']['hashtags'])
+    tweet['hashtags'] = [x['text'] for x in doc['entities']['hashtags']]
     tweet['coordinates'] = doc['coordinates']
     tweet['timestamp_ms'] = doc['timestamp_ms']
     tweet['text'] = doc['text']
RefactoringTool: Files that need to be modified:
RefactoringTool: tweet_utils.py
```

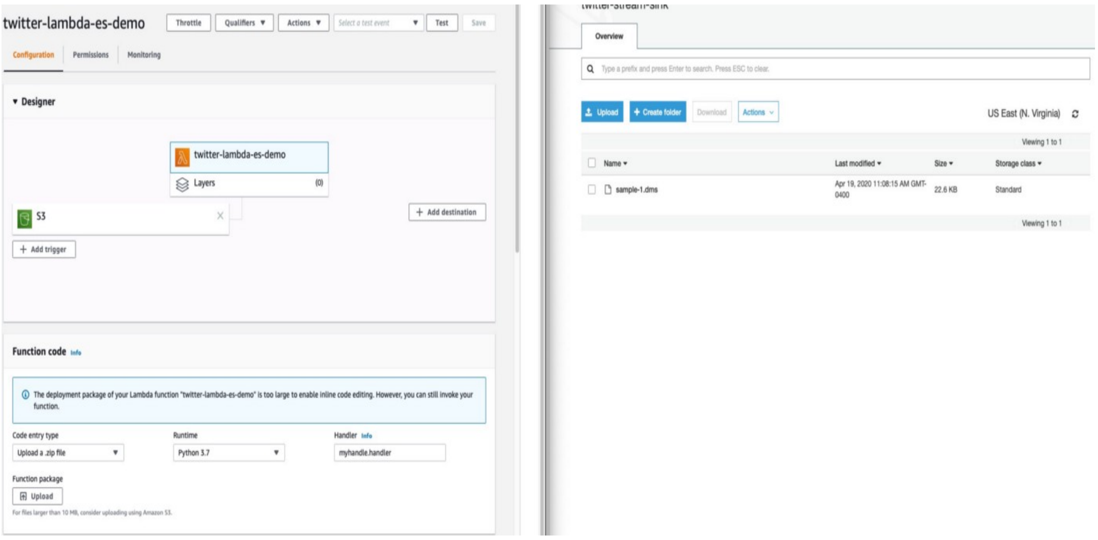
An example of using *2to3 command to port Python 2 code to Python 3.*

The incompatible issues have been fixed:

- Since the release of Elasticsearch 7.0.0, mapping types are removed.
- Elasticsearch's Python Client also experiences changes from previous releases, particularly the usage of the *bulk* method.

To get familiar with the Elasticsearch's Python client, you can open a *Jupiter notebook* to test the connection with the ES cluster.

After looking into the code, we need to package the code. To test the Lambda function, we put a captured twitter file into the S3 bucket, and see if the tweets are parsed correctly and loaded into the ES cluster.



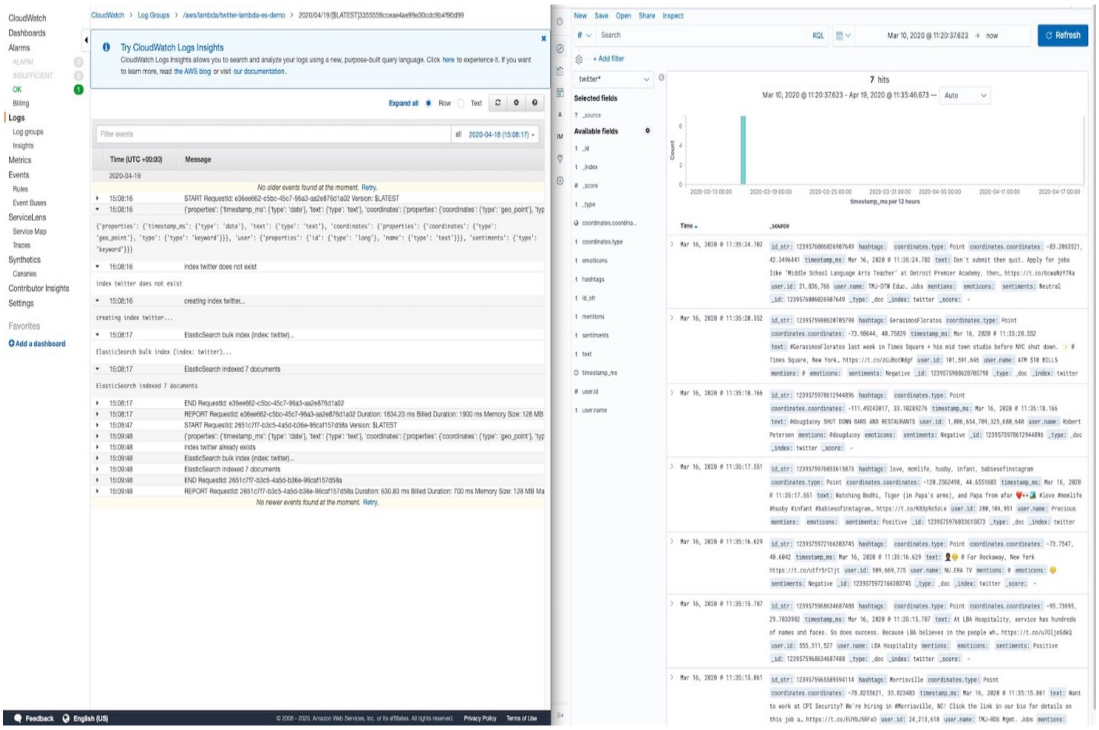
The deployment package of your Lambda function "twitter-lambda-es-demo" is too large to enable inline code editing. However, you can still invoke your function.

Code entry type: Upload a .zip file
Runtime: Python 3.7
Handler: myhandle.handler
Function package: Upload
For files larger than 10 MB, consider uploading using Amazon S3.

□ Name ▾	Last modified ▾	Size ▾	Storage class ▾
□ sample-1.dms	Apr 19, 2020 11:08:15 AM GMT-0400	22.6 KB	Standard

The left window shows the panel of the Lambda function, and the right window shows putting a sample data file with tweets into S3.

If the data is loaded into the ES cluster successfully, we can use Kibana's "Discover" function to check it.



Try CloudWatch Logs Insights
CloudWatch Logs Insights allows you to search and analyze your logs using a new, purpose-built query language. Click here to experience it. If you want to learn more, read the AWS blog or visit our documentation.

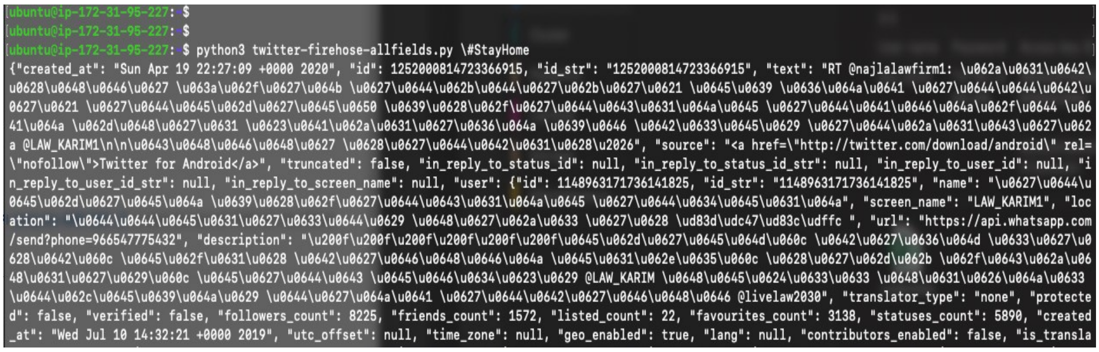
```
Time (UTC +00:00)   Message
2020-04-19
                    No older events found at the moment. Retry.
15:08:16            START RequestId: e36ee662-c5bc-45c7-96a3-aa2e876d1a02 Version: $LATEST
15:08:16            {'properties': {'timestamp_ms': {'type': 'date'}, 'text': {'type': 'text'}, 'coordinates': {'properties': {'coordinates': {'type': 'geo_point'}, 'typ
{'properties': {'timestamp_ms': {'type': 'date'}, 'text': {'type': 'text'}, 'coordinates': {'properties': {'coordinates': {'type':
'geo_point'}, 'type': {'type': 'keyword'}}}, 'user': {'properties': {'id': {'type': 'long'}, 'name': {'type': 'text'}}}, 'sentiments': {'type':
'keyword'}}}
15:08:16            index twitter does not exist
index twitter does not exist
15:08:16            creating index twitter...
creating index twitter...
15:08:17            ElasticSearch bulk index (index: twitter)...
ElasticSearch bulk index (index: twitter)...
15:08:17            ElasticSearch indexed 7 documents
ElasticSearch indexed 7 documents
15:08:17            END RequestId: e36ee662-c5bc-45c7-96a3-aa2e876d1a02
15:08:17            REPORT RequestId: e36ee662-c5bc-45c7-96a3-aa2e876d1a02 Duration: 1834.23 ms Billed Duration: 1900 ms Memory Size: 128 MB
15:09:47            START RequestId: 2651c7f7-b3c5-4a5d-b36e-96caf157d58a Version: $LATEST
15:09:48            {'properties': {'timestamp_ms': {'type': 'date'}, 'text': {'type': 'text'}, 'coordinates': {'properties': {'coordinates': {'type': 'geo_point'}, 'typ
15:09:48            index twitter already exists
15:09:48            ElasticSearch bulk index (index: twitter)...
15:09:48            ElasticSearch indexed 7 documents
15:09:48            END RequestId: 2651c7f7-b3c5-4a5d-b36e-96caf157d58a
15:09:48            REPORT RequestId: 2651c7f7-b3c5-4a5d-b36e-96caf157d58a Duration: 630.83 ms Billed Duration: 700 ms Memory Size: 128 MB Ma
                    No newer events found at the moment. Retry.
```

```
Time ▾                        _source
Mar 16, 2020 @ 11:35:24.702   id_str: 1239576006026907649 hashtags:  coordinates.type: Point coordinates.coordinates: -83.2063521,
                              42.3496441 timestamp_ms: Mar 16, 2020 @ 11:35:24.702 text: Don't submit then quit. Apply for jobs
                              like 'Middle School Language Arts Teacher' at Detroit Premier Academy, then… https://t.co/6cwaNzY7Ra
                              user.id: 21,836,766 user.name: TMJ-DTW Educ. Jobs mentions:  emoticons:  sentiments: Neutral
                              _id: 1239576006026907649 _type: _doc _index: twitter _score: -
Mar 16, 2020 @ 11:35:20.552   id_str: 1239575988620705798 hashtags: GerasimosFloratos coordinates.type: Point
                              coordinates.coordinates: -73.98644, 40.75839 timestamp_ms: Mar 16, 2020 @ 11:35:20.552
                              text: #GerasimosFloratos last week in Times Square + his mid town studio before NYC shut down. ✨ @
                              Times Square, New York… https://t.co/zUJBotWdgF user.id: 101,591,646 user.name: ATM $10 BILLS
                              mentions: @ emoticons:  sentiments: Negative _id: 1239575988620705798 _type: _doc _index: twitter
Mar 16, 2020 @ 11:35:18.166   id_str: 1239575978612944896 hashtags:  coordinates.type: Point
                              coordinates.coordinates: -111.49243017, 33.10289276 timestamp_ms: Mar 16, 2020 @ 11:35:18.166
                              text: @dougducey SHUT DOWN BARS AND RESTAURANTS user.id: 1,006,654,709,325,680,640 user.name: Robert
                              Petersen mentions: @dougducey emoticons:  sentiments: Negative _id: 1239575978612944896 _type: _doc
                              _index: twitter _score: -
Mar 16, 2020 @ 11:35:17.551   id_str: 1239575976033615873 hashtags: love, momlife, husby, infant, babiesofinstagram
                              coordinates.type: Point coordinates.coordinates: -120.2562498, 44.6551603 timestamp_ms: Mar 16, 2020
                              @ 11:35:17.551 text: Watching Bodhi, Tiger (in Papa's arms), and Papa from afar ❤️👀🏞️ #love #momlife
                              #husby #infant #babiesofinstagram… https://t.co/K83p9z5zLe user.id: 200,104,951 user.name: Precious
                              mentions:  emoticons:  sentiments: Positive _id: 1239575976033615873 _type: _doc _index: twitter
Mar 16, 2020 @ 11:35:16.629   id_str: 1239575972166303745 hashtags:  coordinates.type: Point coordinates.coordinates: -73.7547,
                              40.6042 timestamp_ms: Mar 16, 2020 @ 11:35:16.629 text: 🧑🏿😷 @ Far Rockaway, New York
                              https://t.co/utfr5rC1jt user.id: 509,669,775 user.name: NU.ERA TV mentions: @ emoticons: 😷
                              sentiments: Negative _id: 1239575972166303745 _type: _doc _index: twitter _score: -
Mar 16, 2020 @ 11:35:15.787   id_str: 1239575968634687488 hashtags:  coordinates.type: Point coordinates.coordinates: -95.73695,
                              29.7833982 timestamp_ms: Mar 16, 2020 @ 11:35:15.787 text: At LBA Hospitality, service has hundreds
                              of names and faces. So does success. Because LBA believes in the people wh… https://t.co/u7OljoSdkQ
                              user.id: 555,311,527 user.name: LBA Hospitality mentions:  emoticons:  sentiments: Positive
                              _id: 1239575968634687488 _type: _doc _index: twitter _score: -
Mar 16, 2020 @ 11:35:15.061   id_str: 1239575965589594114 hashtags: Morrisville coordinates.type: Point
                              coordinates.coordinates: -78.8255621, 35.823483 timestamp_ms: Mar 16, 2020 @ 11:35:15.061 text: Want
                              to work at CPI Security? We're hiring in #Morrisville, NC! Click the link in our bio for details on
                              this job a… https://t.co/EUYbJ5RFxD user.id: 24,213,618 user.name: TMJ-RDU Mgmt. Jobs mentions:
```

The left window shows the trace logs during invocating the Lambda function, and the right window visualizes the tweets loaded into ES.

Apart from the code changes on the Lambda function, I use a Python program running on an AWS EC2 instance to capture tweets, which can be found here.

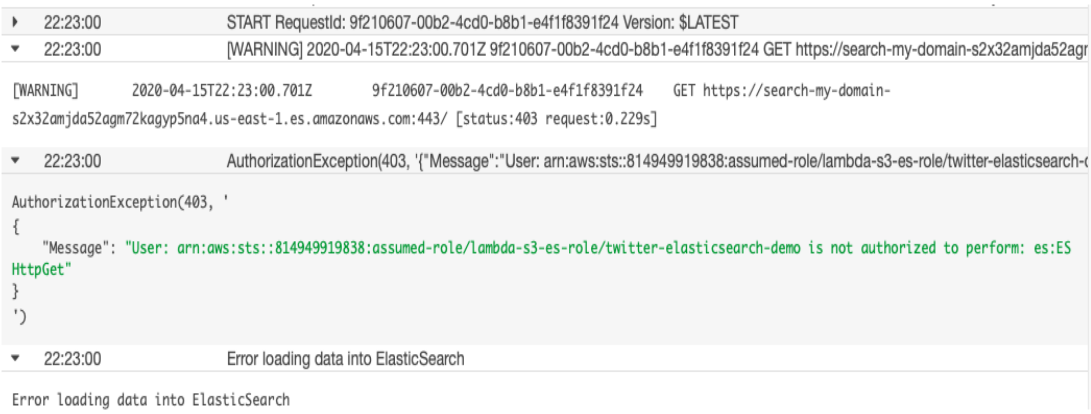
The *reference article* contains a **node.js program** to capture tweets, and either of them does the same job.



```
ubuntu@ip-172-31-95-227:~$
ubuntu@ip-172-31-95-227:~$
ubuntu@ip-172-31-95-227:~$ python3 twitter-firehose-allfields.py \#StayHome
{"created_at": "Sun Apr 19 22:27:09 +0000 2020", "id": 1252000814723366915, "id_str": "1252000814723366915", "text": "RT @najlalawfirm1: \u062a\u0631\u0642\
u0628\u0648\u0646\u0627 \u063a\u062f\u0627\u064b \u0627\u0644\u062b\u0644\u0627\u062b\u0627\u0621 \u0645\u0639 \u0636\u064a\u0641 \u0627\u0644\u0644\u0642\u
0627\u0621 \u0627\u0644\u0645\u062d\u0627\u0645\u0650 \u0639\u0628\u062f\u0627\u0644\u0643\u0631\u064a\u0645 \u0627\u0644\u0641\u0646\u064a\u062f\u0644 \u06
41\u064a \u062d\u0648\u0627\u0631 \u0623\u0641\u062a\u0631\u0627\u0636\u064a \u0639\u0646 \u0642\u0633\u0645\u0629 \u0627\u0644\u062a\u0631\u0643\u0627\u062
a @LAW_KARIM1\n\n\u0643\u0648\u0646\u0648\u0627 \u0628\u0627\u0644\u0642\u0631\u0628\u2026", "source": "<a href=\"http://twitter.com/download/android\" rel=
\"nofollow\">Twitter for Android</a>", "truncated": false, "in_reply_to_status_id": null, "in_reply_to_status_id_str": null, "in_reply_to_user_id": null, "i
n_reply_to_user_id_str": null, "in_reply_to_screen_name": null, "user": {"id": 1148963171736141825, "id_str": "1148963171736141825", "name": "\u0627\u0644\u
0645\u062d\u0627\u0645\u064a \u0639\u0628\u062f\u0627\u0644\u0643\u0631\u064a\u0645 \u0627\u0644\u0634\u0645\u0631\u064a", "screen_name": "LAW_KARIM1", "loc
ation": "\u0644\u0644\u0645\u0631\u0627\u0633\u0644\u0629 \u0648\u0627\u062a\u0633 \u0627\u0628 \ud83d\udc47\ud83c\udffc ", "url": "https://api.whatsapp.com
/send?phone=966547775432", "description": "\u200f\u200f\u200f\u200f\u200f\u200f\u0645\u062d\u0627\u0645\u064d\u060c \u0642\u0627\u0636\u064d \u0633\u0627\u0
628\u0642\u060c \u0645\u062f\u0631\u0628 \u0642\u0627\u0646\u0648\u0646\u064a \u0645\u0631\u062e\u0635\u060c \u0628\u0627\u062d\u062b \u062f\u0643\u062a\u06
48\u0631\u0627\u0629\u060c \u0645\u0627\u0644\u0643 \u0645\u0646\u0634\u0623\u0629 @LAW_KARIM \u0648\u0645\u0624\u0633\u0633 \u0648\u0631\u0626\u064a\u0633
\u0644\u062c\u0645\u0639\u064a\u0629 \u0644\u0627\u064a\u0641 \u0627\u0644\u0642\u0627\u0646\u0648\u0646 @livelaw2030", "translator_type": "none", "protecte
d": false, "verified": false, "followers_count": 8225, "friends_count": 1572, "listed_count": 22, "favourites_count": 3138, "statuses_count": 5890, "created
_at": "Wed Jul 10 14:32:21 +0000 2019", "utc_offset": null, "time_zone": null, "geo_enabled": true, "lang": null, "contributors_enabled": false, "is_transla
```

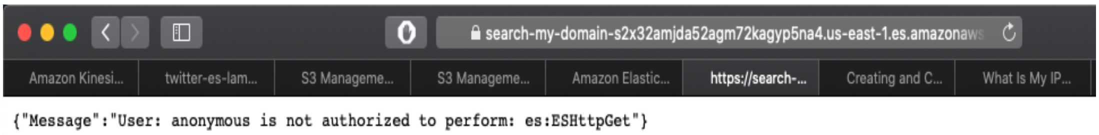
Running the Python tweet capture program

One thing to note here. When I implemented the pipeline and tried to test to load data into the ES cluster, I met authentication errors on Lambda invocation and Kibana:



```
▸ 22:23:00    START RequestId: 9f210607-00b2-4cd0-b8b1-e4f1f8391f24 Version: $LATEST
▾ 22:23:00    [WARNING] 2020-04-15T22:23:00.701Z 9f210607-00b2-4cd0-b8b1-e4f1f8391f24 GET https://search-my-domain-s2x32amjda52agr

[WARNING]    2020-04-15T22:23:00.701Z    9f210607-00b2-4cd0-b8b1-e4f1f8391f24    GET https://search-my-domain-
s2x32amjda52agm72kagyp5na4.us-east-1.es.amazonaws.com:443/ [status:403 request:0.229s]

▾ 22:23:00    AuthorizationException(403, '{"Message":"User: arn:aws:sts::814949919838:assumed-role/lambda-s3-es-role/twitter-elasticsearch-d

AuthorizationException(403, '
{
    "Message": "User: arn:aws:sts::814949919838:assumed-role/lambda-s3-es-role/twitter-elasticsearch-demo is not authorized to perform: es:ES
HttpGet"
}
')

▾ 22:23:00    Error loading data into ElasticSearch

Error loading data into ElasticSearch
```

When I put a data file into the S3 bucket, the Lambda function reported such an error.



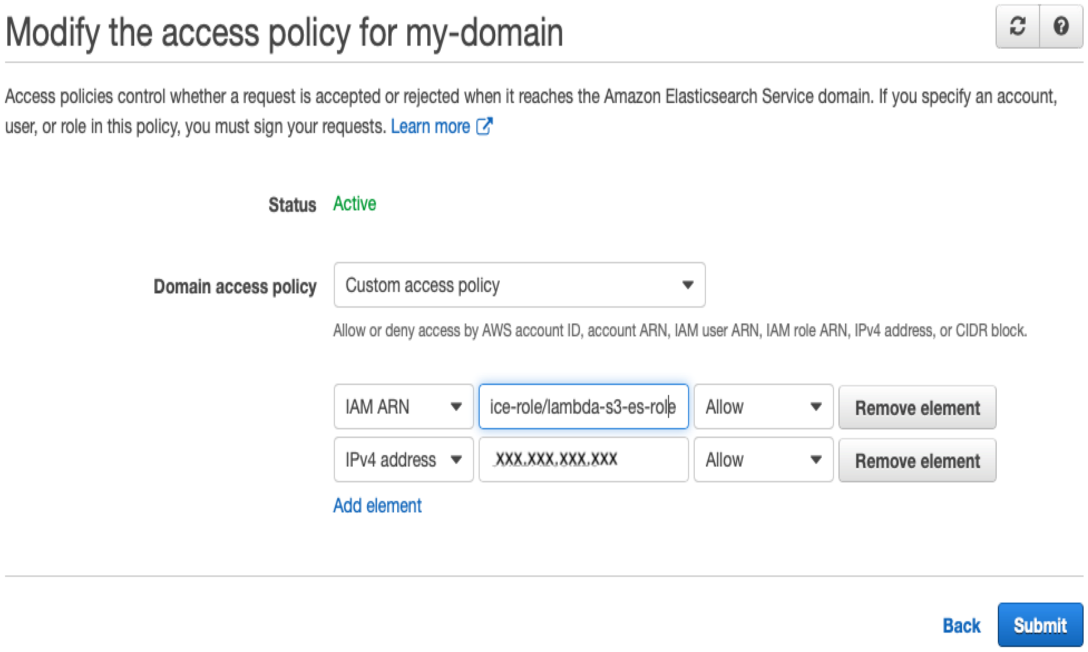
```
{"Message":"User: anonymous is not authorized to perform: es:ESHttpGet"}
```

When I tried to access Kibana after setting up the ES cluster.

To find out the reason, we need to go to the page specifying the **Identity and Access Management** of Amazon ES.

> **The primary appeal of IP-based policies is that they allow unsigned requests to an Amazon ES domain, which lets you use clients like curl and Kibana or access the domain through a proxy server.**

All requests to the Amazon ES configuration API must be signed. In the above errors, even though the Lambda execution role was added, the requests were unsigned and rejected, especially HTTP requests. To resolve this problem, we can add IP-based policies, or adding signs using **AWS SDK or requests**. So we need to add an IP-based policy into the ES access policies:



Modify the access policy for my-domain

Access policies control whether a request is accepted or rejected when it reaches the Amazon Elasticsearch Service domain. If you specify an account, user, or role in this policy, you must sign your requests. Learn more

Status Active

Domain access policy: Custom access policy

Allow or deny access by AWS account ID, account ARN, IAM user ARN, IAM role ARN, IPv4 address, or CIDR block.

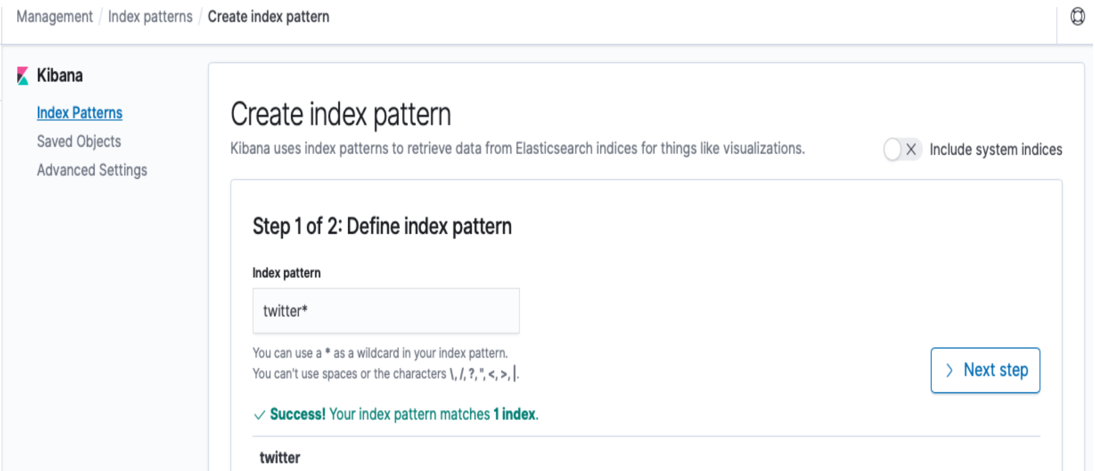
IAM ARN | ice-role/lambda-s3-es-role | Allow | Remove element

IPv4 address | XXX.XXX.XXX.XXX | Allow | Remove element

Kibana visualization and Twitter analysis

Once we start running the twitter capture program, loads of tweets would be flown into S3 and the Lambda function would handle the data files. The best way to reflect the data is through the visualization utility *Kibana*, provided in Elasticsearch.

All data loaded into Elasticsearch need to be assigned with indices, and thus Kibana can use *index patterns* to retrieve the data. In *twitter_to_es.py*, tweets are indexed with “twitter”. Now we can create an index pattern “twitter*” and start discovering the data in Kibana.



Management / Index patterns / Create index pattern

Create index pattern

Kibana uses index patterns to retrieve data from Elasticsearch indices for things like visualizations.

Include system indices

Step 1 of 2: Define index pattern

Index pattern: twitter*

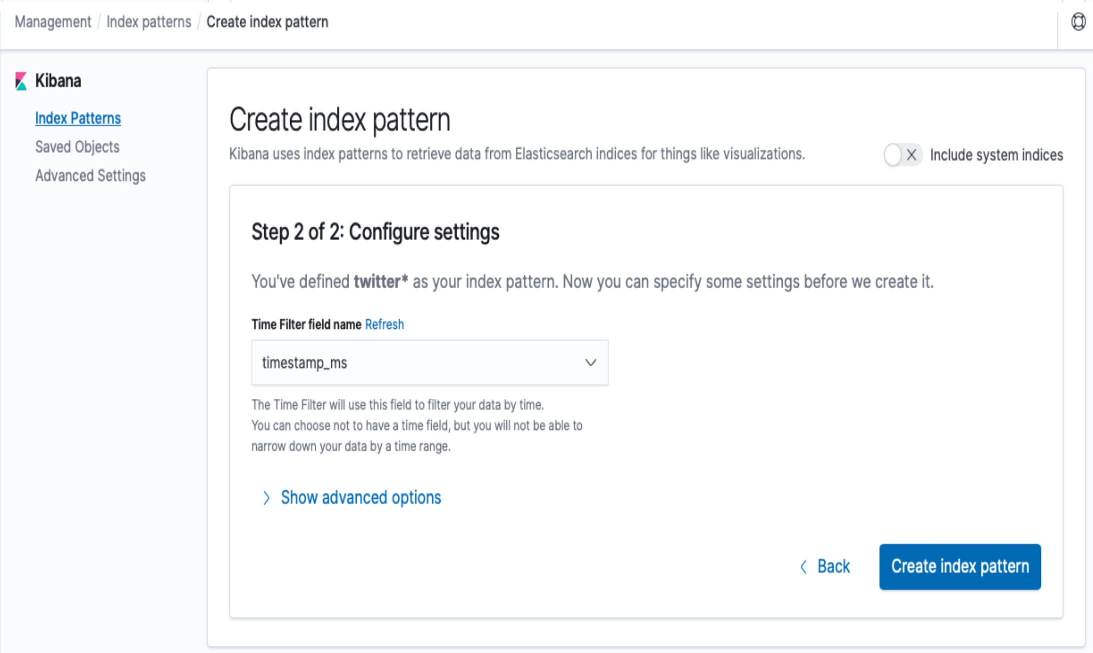
You can use a * as a wildcard in your index pattern.
You can't use spaces or the characters \, /, ?, ", <, >, |.

✓ Success! Your index pattern matches 1 index.

twitter

Create the index pattern “twitter*” to match all indices starting with “twitter”.

Rows per page: 10



Create index pattern

Kibana uses index patterns to retrieve data from Elasticsearch indices for things like visualizations.

Step 2 of 2: Configure settings

You've defined **twitter*** as your index pattern. Now you can specify some settings before we create it.

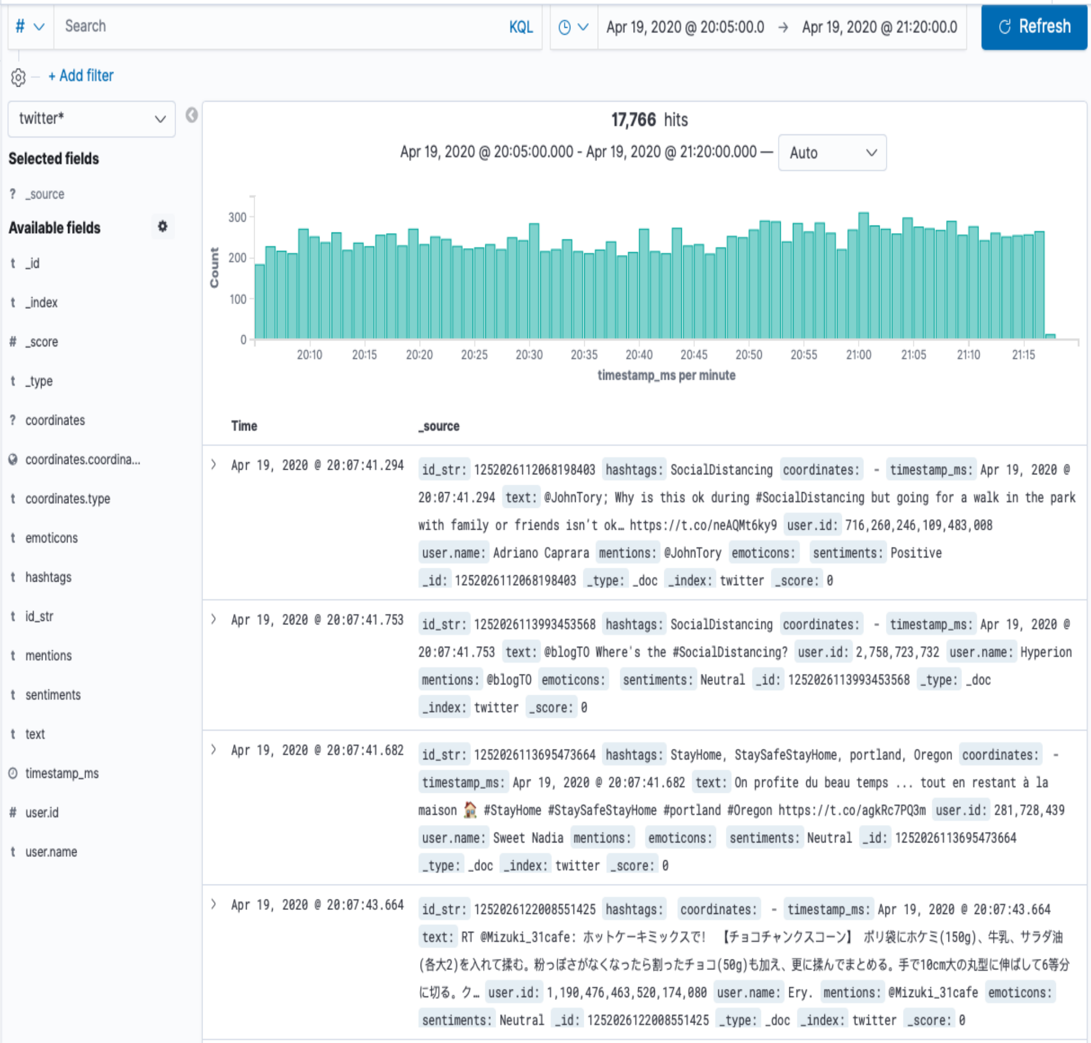
Time Filter field name Refresh

timestamp_ms

The Time Filter will use this field to filter your data by time. You can choose not to have a time field, but you will not be able to narrow down your data by a time range.

The “timestamp_ms” field is specified when we add the mapping in tweet_utils.py.

After the creation of the index pattern, we can explore the data by choosing the “Discover” button on the left-sidebar and the data can be presented in time series.



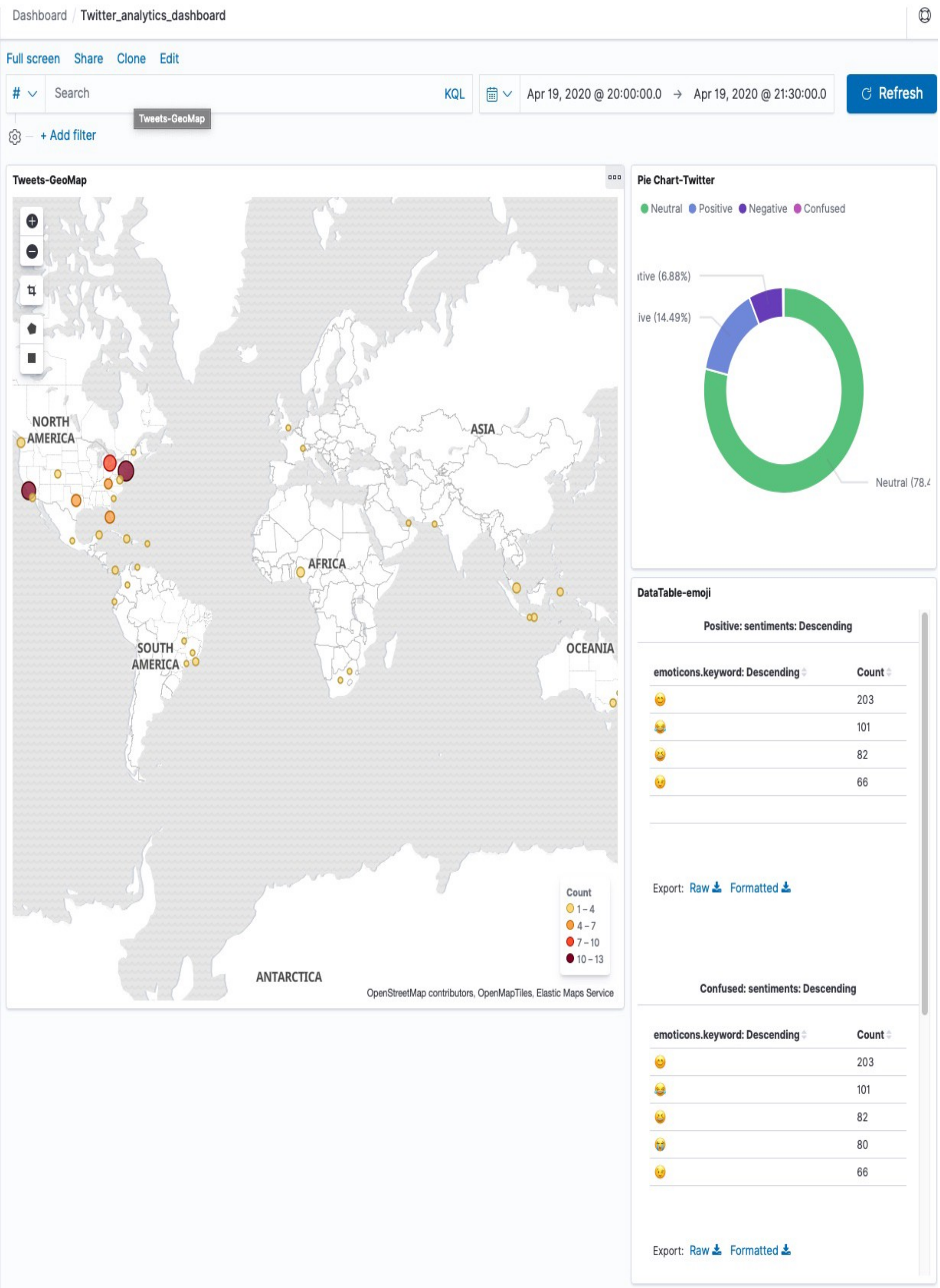
17,766 hits

Apr 19, 2020 @ 20:05:00.000 - Apr 19, 2020 @ 21:20:00.000 — Auto

Time	_source
Apr 19, 2020 @ 20:07:41.294	id_str: 1252026112068198403 hashtags: SocialDistancing coordinates: - timestamp_ms: Apr 19, 2020 @ 20:07:41.294 text: @JohnTory; Why is this ok during #SocialDistancing but going for a walk in the park with family or friends isn't ok… https://t.co/neAQMt6ky9 user.id: 716,260,246,109,483,008 user.name: Adriano Caprara mentions: @JohnTory emoticons: sentiments: Positive _id: 1252026112068198403 _type: _doc _index: twitter _score: 0
Apr 19, 2020 @ 20:07:41.753	id_str: 1252026113993453568 hashtags: SocialDistancing coordinates: - timestamp_ms: Apr 19, 2020 @ 20:07:41.753 text: @blogTO Where's the #SocialDistancing? user.id: 2,758,723,732 user.name: Hyperion mentions: @blogTO emoticons: sentiments: Neutral _id: 1252026113993453568 _type: _doc _index: twitter _score: 0
Apr 19, 2020 @ 20:07:41.682	id_str: 1252026113695473664 hashtags: StayHome, StaySafeStayHome, portland, Oregon coordinates: - timestamp_ms: Apr 19, 2020 @ 20:07:41.682 text: On profite du beau temps ... tout en restant à la maison 🏠 #StayHome #StaySafeStayHome #portland #Oregon https://t.co/agkRc7PQ3m user.id: 281,728,439 user.name: Sweet Nadia mentions: emoticons: sentiments: Neutral _id: 1252026113695473664 _type: _doc _index: twitter _score: 0
Apr 19, 2020 @ 20:07:43.664	id_str: 1252026122008551425 hashtags: coordinates: - timestamp_ms: Apr 19, 2020 @ 20:07:43.664 text: RT @Mizuki_31cafe: ホットケーキミックスで! 【チョコチャンクスコーン】 ポリ袋にホケミ(150g)、牛乳、サラダ油(各大2)を入れて揉む。粉っぽさがなくなったら割ったチョコ(50g)も加え、更に揉んでまとめる。手で10cm大の丸型に伸ばして6等分に切る。ク… user.id: 1,190,476,463,520,174,080 user.name: Ery. mentions: @Mizuki_31cafe emoticons: sentiments: Neutral _id: 1252026122008551425 _type: _doc _index: twitter _score: 0

In this article, I pick the hashtags “**#StayHome**” and “**#SocialDistancing**” to mine Twitter. Like the *reference article*, I create a dashboard to visualize the tweets. There are three visualizations in the dashboard:

- A coordinate map to demonstrate the geographical distribution of the tweets, only valid if tweets contain location information.
- A pie chart to demonstrate sentiment popularity when users send tweets, including three types of emotions, positive, neutral and negative.
- A data table to count the emojis contained in each sentiment, and only the top-5 emojis are listed.



Dashboard / Twitter_analytics_dashboard

DataTable-emoji

Positive: sentiments: Descending

emoticons.keyword: Descending	Count
😊	203
😂	101
😆	82
😉	66

Export: Raw Formatted

Confused: sentiments: Descending

emoticons.keyword: Descending	Count
😊	203
😂	101
😆	82
😭	80
😉	66

Export: Raw Formatted

The dashboard created in Kibana.

Conclusion

AWS Lambda and Elasticsearch are pretty powerful technologies and this post may just demonstrate one case among the application scenarios. Besides real-time data processing, Lambda can be integrated with ETL (Extract, Transform, Load) and application backends. Elasticsearch has established a reputation on logging/log analysis and full-text searching.

I hope you can find fun when done the reading and being able to fiddle with the big data technologies. This is how big data fascinates me.